Amendments to the Claims

Claim 1 (currently amended): A computer-implemented method of programmatically building queries, comprising steps of:

programmatically building a query user interface to query a content source, wherein the query user interface comprises a plurality of query parameters, each query parameter comprising a unique query parameter name, a query qualifier, and a query parameter value, further comprising:

dynamically identifying the content source to be queried;

programmatically determining a plurality of content values specified in the dynamically-identified content source;

programmatically determining, based on the specified content values, a plurality of content types corresponding thereto;

~~programmatically identifying, for a content source, at least one element thereof by programmatically obtaining one or more tag names from a markup language document used for rendering a view of the content source;~~

using at least one of the ~~programmatically-obtained~~ programmatically-determined ~~tag names~~ content types to consult a lookup component to obtain at least [[one]] two ~~candidate~~ query parameter [[name]] names usable to query ~~for querying~~ the content source;

programmatically identifying, for each of the obtained query parameter names, at least one query qualifier corresponding thereto, each query qualifier usable in determining a match when comparing selected ones of the content values to that query parameter name;

programmatically identifying, for at least one of the obtained query parameter names, at least one value usable therewith as a query parameter value;

programmatically building the plurality of query parameters by associating, with each of the obtained query parameter names, the programmatically-identified at least one query qualifier corresponding thereto and the programmatically-identified at least one value usable therewith as a query parameter value, if any; and

displaying on the query user interface, for each of the programmatically-built query parameters, the obtained ~~candidate~~ query parameter name, a first selector usable to select one of the at least one query qualifiers corresponding thereto, and a second selector usable to select at least one of the at least one values usable therewith, if any, or for providing at least one user-entered value usable therewith; and ~~on a user interface display configured~~

enabling a user to build a query command~~, responsive to selection by a user of at least one of the displayed candidate query parameter name or names,~~ to query the content source by using, for each of at least one of the displayed query parameter names, the first selector to select one of the associated query qualifiers and using the second selector to select at least one of: (1) at least one of the associated values, if any, or (2) at least one user-entered value.

Claim 2 (canceled)

Claim 3 (previously presented): The method according to Claim 1, wherein the using step further comprises using information regarding the user when consulting the lookup component.

Claim 4 (currently amended): The method according to Claim 1, further comprising the step of:

programmatically identifying at least one query extension parameter [[name]] for the query command, responsive to a request from the user to extend the ~~query command~~ display on the query user interface, further comprising, for each of the at least one query extension parameters:

using at least one of the obtained query parameter names to obtain a related query parameter name;

programmatically identifying at least one query qualifier corresponding to the obtained related query parameter name, each query qualifier usable in determining a match when comparing selected ones of the content values to the obtained related query parameter name; and

programmatically building the query extension parameter by associating, with the obtained related query parameter name, the programmatically-identified at least one query qualifier corresponding thereto; and

wherein the displaying step further comprises also displaying each of the at least one programmatically-identified query extension ~~parameter name or names~~ parameters as additional ones of the ~~candidate~~ programmatically-built query ~~parameter names~~ parameters.

Claim 5 (canceled)

Claim 6 (previously presented): The method according to Claim 1, wherein the using step further comprises using information regarding the content source when consulting the lookup component.

Claim 7 (previously presented): The method according to Claim 3, wherein the information regarding the user comprises at least one of: a role of the user, preferences of the user, a device used by the user, or an identification of the user.

Claims 8 - 23 (canceled)

Claim 24 (new): A system configured to programmatically build queries, comprising:

means for programmatically building a query user interface to query a content source, wherein the query user interface comprises a plurality of query parameters, each query parameter comprising a unique query parameter name, a query qualifier, and a query parameter value, further comprising:

means for dynamically identifying the content source to be queried;

means for programmatically determining a plurality of content values specified in the dynamically-identified content source;

means for programmatically determining, based on the specified content values, a plurality of content types corresponding thereto;

means for using at least one of the programmatically-determined content types to consult a lookup component to obtain at least two query parameter names usable to query the content source;

means for programmatically identifying, for each of the obtained query parameter names, at least one query qualifier corresponding thereto, each query qualifier usable in determining a match when comparing selected ones of the content values to that query parameter name;

means for programmatically identifying, for at least one of the obtained query parameter names, at least one value usable therewith as a query parameter value;

means for programmatically building the plurality of query parameters by associating, with each of the obtained query parameter names, the programmatically-identified at least one query qualifier corresponding thereto and the programmatically-identified at least one value usable therewith as a query parameter value, if any; and

means for displaying on the query user interface, for each of the programmatically-built query parameters, the obtained query parameter name, a first selector usable to select one of the at least one query qualifiers corresponding thereto, and a second selector usable to select at least one of the at least one values usable therewith, if any, or for providing at least one user-entered value usable therewith; and

means for enabling a user to build a query command to query the content source by using, for each of at least one of the displayed query parameter names, the first selector to select one of the associated query qualifiers and using the second selector to select at least one of: (1) at least one of the associated values, if any, or (2) at least one user-entered value.

Claim 25 (new): A computer program product configured to programmatically build queries, the computer program product embodied on one or more computer-readable storage media and comprising:

computer-readable program code for programmatically building a query user interface to query a content source, wherein the query user interface comprises a plurality of query parameters, each query parameter comprising a unique query parameter name, a query qualifier, and a query parameter value, further comprising:

computer-readable program code for dynamically identifying the content source to be queried;

computer-readable program code for programmatically determining a plurality of content values specified in the dynamically-identified content source;

programmatically determining, based on the specified content values, a plurality of content types corresponding thereto;

computer-readable program code for using at least one of the programmatically-determined content types to consult a lookup component to obtain at least two query parameter names usable to query the content source;

computer-readable program code for programmatically identifying, for each of the obtained query parameter names, at least one query qualifier corresponding thereto, each query qualifier usable in determining a match when comparing selected ones of the content values to that query parameter name;

computer-readable program code for programmatically identifying, for at least one of the obtained query parameter names, at least one value usable therewith as a query parameter value;

computer-readable program code for programmatically building the plurality of query parameters by associating, with each of the obtained query parameter names, the programmatically-identified at least one query qualifier corresponding thereto and the programmatically-identified at least one value usable therewith as a query parameter value, if any; and

computer-readable program code for displaying on the query user interface, for each of the programmatically-built query parameters, the obtained query parameter name, a first selector usable to select one of the at least one query qualifiers corresponding thereto, and a second selector usable to select at least one of the at least one values usable therewith, if any, or for providing at least one user-entered value usable therewith; and

computer-readable program code for enabling a user to build a query command to query the content source by using, for each of at least one of the displayed query parameter names, the first selector to select one of the associated query qualifiers and using the second selector to select at least one of: (1) at least one of the associated values, if any, or (2) at least one user-entered value.